HOCHSCHULE TRIER
Trier University of Applied Sciences
Informatik - Computer Science

Mobile Anwendung für die Kostenschätzung mit Android

Mobile Application for cost estimations in Android

Oliver Fries

Bachelor-Abschlussarbeit

Betreuer: Prof. Dr. Georg Rock

Trier, 29.02.2016

Abstract

Die Wichtigkeit von Kostenschätzungen in IT-Projekten steigt durch immer komplexere und größere Projekte stetig an. Schwierigkeiten bei der Kostenschätzung liegen im Zugriff auf bereits vorhandene Schätzungen, die zur Verbesserung der Schätzwerte beitragen, sowie der Auswahl relevanter Projekte, damit nicht alle verfügbaren betrachtet werden müssen. Diese Arbeit zielt darauf ab den Prozess der Kostenschätzung als mobile Anwendung umzusetzen und die genannten Probleme mit einem dynamischeren Prozess schneller durchführen zu können. Hierbei steht die Umsetzung des Function Point Verfahrens im Fokus, sowie die Möglichkeit mit einem Vergleich der Projekte auf vorangegangene Schätzungen zuzugreifen. Diese mobile Anwendung wurde als Android Applikation umgesetzt und trägt den Namen MobileEstimate. Es ergeben sich dadurch neue Möglichkeiten für Projektmanager und Projektteams zur schnelleren und einfacheren Kostenanalyse bei IT Projekten und zum Monitoring der Projektkosten.

The importance of cost estimation in IT projects is constantly increasing through more complex and larger projects. Some of the difficulties in cost estimations are the access to existing estimations, which help to improve the estimation, and the selection of relevant projects in order to not consider all available projects. This paper aims to implement the cost estimation as a mobile application with a more dynamic process to approach the specified problems. Implementation of the Function Point method is the focus of this paper and the possibility to compare projects and get access to their cost estimation. The mobile application was implemented as an Android application and is called MobileEstimate. This results in new opportunities for project managers and project teams for faster and easier cost analysis for IT projects and the monitoring of project costs.

Contents

1

Introduction

Most of the contracts IT companies subscribe are projects and these are notorious for going past their deadline and over their budget. According to the study of Capgemini in 2014 [?], the importance of cost estimation increases every year. The study asked for the most important requirements in the IT for the next years, with the top requirement to increase the efficiency, which means to lower the costs and to meet determined deadlines. This will enhance the effort companies have to take in planning their projects.

All businesses want to lower the risk of delayed or canceled projects. This results in more effort IT companies have to take in requirements engineering and cost estimation to give their clients an accurate estimation of the upcoming project. As a result of the increase in requirements engineering and, subsequently, the cost estimation, to 6% till 12% the project time will lower to cost overrun to a maximum of 50% [?].

Therefore cost estimation is an important element for planning software projects and can be responsible for successful or failed projects. It is even more important to estimate as precisely as possible to guide the project to success. There are several methods for these estimations that can be used at different phases of the project. These methods of estimation lean their result on the information they get from the development process and the artifacts of the particular project phase. These include requirement documents, diagrams or the program code itself. All available artifacts are depending on the used process model and the project phase [OH11]. Based on the described information a categorization for the actual project can be made, to find the “best fitting” estimation method for the current estimation. These methods can be time-consuming and related projects can most times only be found in the own company context or are based on experiences.

Objectives of the proposal is to develop a mobile application which support the function point estimation. The implemented process aims to makes the estimation process in IT-projects simpler and more efficient. To achieve a better way for estimating costs the most important design guideline was ”Only show what I need when I need it” [?]. The comparison between projects has to be formalized and implemented. This should give the user an overview over terminated projects and how they were estimated. Either to transfer the estimation to the new project or to get a quick view how much days it took.

The application MobileEstimate proves that cost estimation on mobile devices is possible. Its possible to estimate the costs of a project with function point method and among the existing projects related projects can be displayed. Estimation results from a related project can be viewed in the application and transfered to another estimation. Also a complete estimation can be exported as an excel file and processed afterwards.

From an computer scientist viewpoint, the conclusion to be drawn with the implemented application with the Android Design Principles is possible and allows a simpler way to estimate the costs of IT projects. To make the application marketable, plenty still remains to be done and the use of the application in the module "Spezifikation interaktiver Systeme" will give more feedback about the application and what additional features are needed.

2

Theoretical Background

This chapter describes the fundamentals of this paper. The cost estimation process in IT projects and the different methods to calculate the cost of a project are the essential part. The state of the art report combined with the market description will give a short overview over the situation about software estimation tools on the market and the possibility of a mobile solution of cost estimations. Android as the chosen platform and Java as the programming language will not be described in detail here and are assumed to be known.

2.1 Cost estimation in software engineering

The most expensive components of computer systems are software products. While private clients are mostly interested in the final price of the product, business clients of IT companies typically want to know the costs of the software before project launch. As analyzed in the IT-Trends study from Capgemini [?] the IT budget of companies is growing up to 10% every year. Whether developing a new project or standardizing existing software, the project costs are always one of the three key objectives. As human resources are the biggest part of software costs, project managers and especially the business clients want to know the estimated spendings and completion time of a project. Most of the estimation methods focus on this aspect and give the result in man days. These estimated days can then be converted into the real costs.
Basically the cost estimation in software engineering wants to answer the following questions:

1. How much effort is required to complete the project?
2. How much days are needed to complete the project?
3. What is the total cost of the project?

While the projects are a living thing, the effort may change due to unexpected difficulties. For a precise estimation of the total costs an adjustment cannot be avoided and it can be useful to change the estimation method in a later project phase. This means that the estimation process is not an one-time thing but will change through the project.

2.1.1 Estimation Process

It is common to create a first cost estimation before the system design, but also for monitoring purposes, milestones or if the client wants an overview of the project. Each time an actual cots estimation is needed the estimation process is executed which is a set of techniques and procedures that is used to derive the software cost estimate. Kathleen Peters described the basic process of an estimation as it is common in the industry [?].

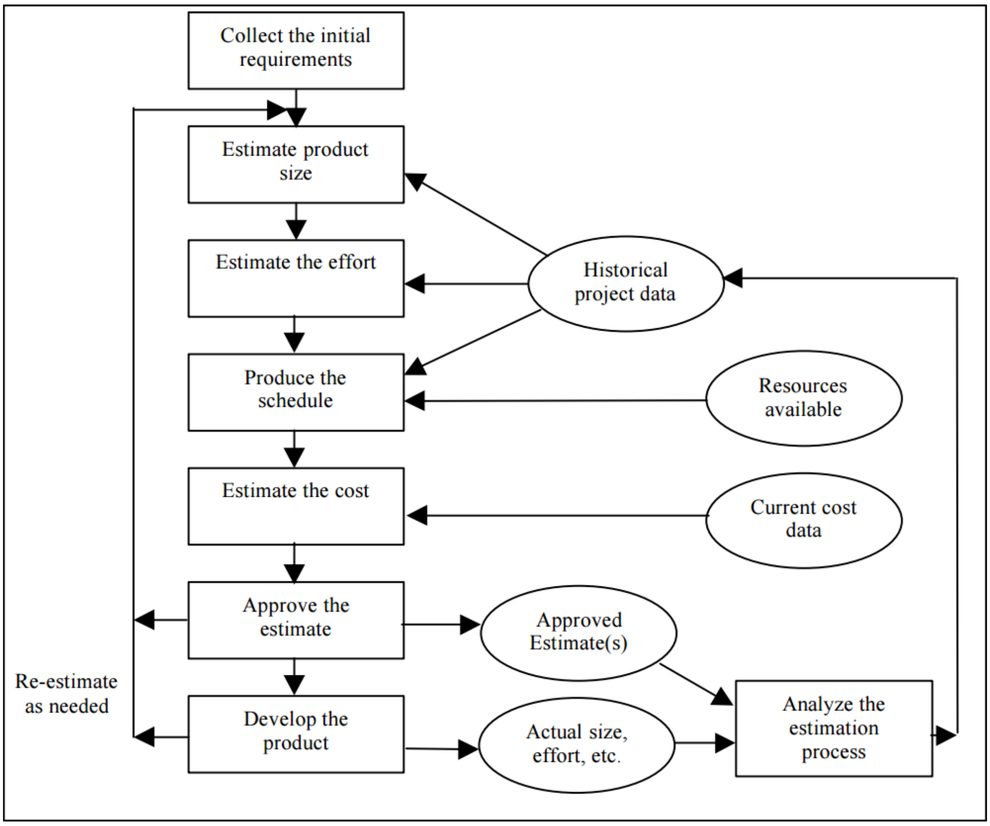

Fig. 2.1. - The Basic Project Estimation Process
Source: Peters, Kathleen - Software Project Estimation, Page 3

As can bee seen from figure 2.1, there are mainly seven steps in the estimation process. The first part is to collect the initial requirements which is essential to know what the project is about evaluate the approximate size. With an selected estimation method, which are described in 2.1.3, the evaluated size of the project is then estimated. Afterwards the effort in man-days is calculated from which the cost schedule is created. To this process can project data from older projects be included in the evaluation. In the process step to approve the estimation, it comes to the decision, if the costs are acceptable or if there have to be adjustments in the range of functions and the re-estimation has to bee started. If the cost estimation is acceptable the development of the product can start or continue.

In this classical view of the estimation process, it will generate three outputs - size, efforts, duration and loading. The output can be described as following:

1. Actual Size - the size of the project in a numerical value to make the project comparable.
2. Manpower Loading - number of personnel that are allocated to the project as a function of time.
3. Project Duration - time that is needed to complete the project.
4. Effort - amount of effort required to complete the project and is usually measured in units as man-days (MD) or person-months (PM).

As described before, the estimation process can be triggered at any time in the project to re-estimate the costs. Depending on the project stage another estimation method then used before can be better.

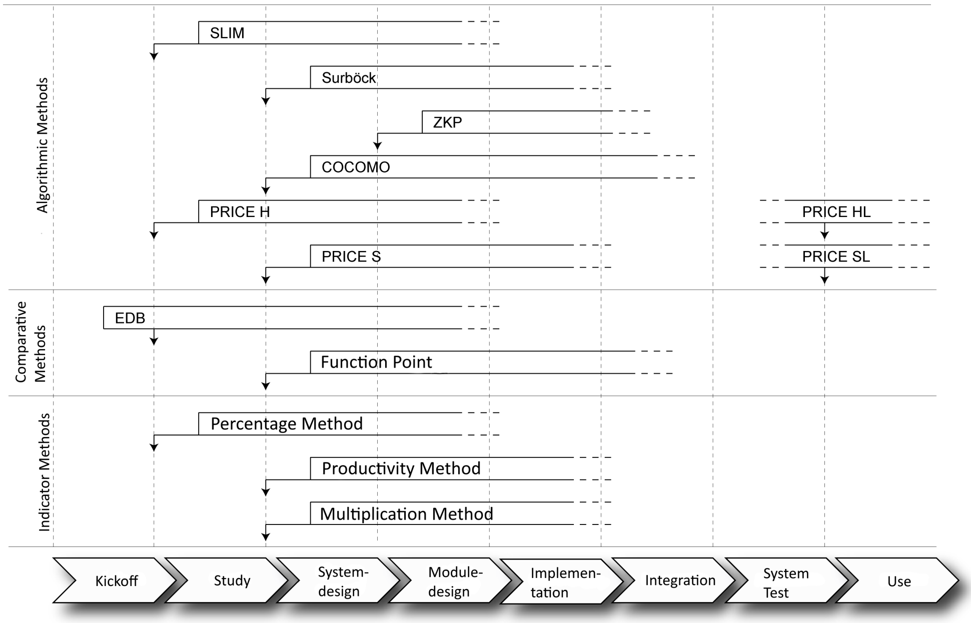

Fig. 2.2. - Starting Points of Estimation Methods
Source: `http://winfwiki.wi-fom.de`

As you can see in figure 2.2, the SLIM method for example is more suitable at the beginning of a project, whereas the ZKP method is more suitable after the system design. Most of the estimation methods can be used after the study, this is because then an rough overview of the project size exists. Different estimation methods may also change the evaluation output, which is one of the difficulties of cost estimations.

2.1.2 Difficulty of estimations

One of the problems in estimating costs is, that the implementation is only a small part of the project. Beside the project planning are many administrative tasks to do, coordination of the project, searching and fixing errors. The most estimation methods evaluate the estimated time for the implementation and too little or no time is evaluated for the non implementation tasks. This resolves in an over- or underestimation of the non implementation tasks [**?**].
Most project managers rely on their experience from older projects, which is an advantage but is most times overrated. As the technology changes fast and new projects inherit new problems there is no prior experience for some parts of the project. Another reason for this is the special feature of projects, their unique nature. The more positive outlook of people is the reason for mostly underestimated costs with estimations where participants of the project are inherited. Customers often got a target time for the project, which leads in adjustment of the cost estimation. This leads to budget overrun if thee needed resources are not available for the project [**?**].
It is also not guaranteed that the estimated costs are accurate and stay within the budget. An estimation can easily go past their estimated target as new technologies and unexpected difficulties are commonplace. A partial requirements engineering can also cause an inaccurate cost estimation due to unexpected difficulties in the implementation. However requirements engineering is not the main topic of this paper but cost estimations. To this are different methodical approaches possible in which estimation methods can be classified.

2.1.3 Methods for estimation

Estimation methods are different metrics to calculate the costs of a project and there are different approaches to categorize the estimation methods in literature. This categorization is based on the book ”Management von IT-Projekten” from Hans W. Wieczorrek, where the methods are subdivided in algorithmic, comparative, operating figures and the expert discussion [**?**].
All estimation techniques have in common that only with a combined use of these different estimation methods a suitable result can be achieved as a measured value for the project. To evaluate the needed effort is determined with the underlying metric. On the basis of charge rates the effort size is calculated out of it.

Algorithmic Method

The algorithmic method calculates the estimated effort with a closed formula, which is based on empiric evaluation of effort for already terminated projects or on existing mathematically models. Different forms of this methods are the weight and the sampling method, which only differ in their usage.
The accuracy of estimated efforts depend primarily on the precision of the influence

factors [?]. The algorithmic method always connects measurable project sizes, such as lines of code and implementable features, with influencing factors to get the result as required effort in staff and time. The basic formula for this calculation is:

$$\boxed{\textit{Personnel costs = f(result quantity, influencing factors)}\ [?]}$$

Comparative Method

Not based on a formula or numerical connection, the comparative method tries to create a reference between the planned and past projects. Therefor projects from the own company or the same industry sector are analyzed with appropriate comparison methods. This estimation method has the advantage, that it can be used early in development of the project [?]. These method can be used for hardware and for software projects.

Key Figures Method

Estimation methods based on key figures can be differed to multiplier and percentage method. The multiplier method uses units of power as the base to estimate the total expenditure, whereas the percentage method uses the effort of a project stage to estimate the effort for the next stage.
After completion of projects a post calculation determine the total project costs and the amount of specific types of costs. In order ro calculate these costs, they will be divided by the scope of the developed product. This results in new key figures which can be used for new projects by multiplication of the estimated scope with the appropriate key figures.
Regular actualization of the key figures is necessary for right results and because these key figures are the base of the estimation [?].

Expert Discussion Method

As a quantitative, heuristic method uses the expert discussion knowledge from selected groups of people. It differentiate between for kinds: the single person or multiple interview, Delphi method and the assessment meeting.
The advantage of this estimation method is, that they are useful for all project types but experiences with the expert discussion shows, that this method is strongly affected by subjective opinions and the experience of the interviewees. As a result, expert discussions should never be used for complete projects but only for sub projects [?].

2.2 Estimation Techniques

The existing estimation techniques rely on experience-based judgments by project managers who created, by combining estimation methods, techniques someone can

use to estimate the effort of a project. Most of the estimation techniques became popular in the 80's but with the agile projects more estimation techniques for these project types arise.
Because of the fundamental uniqueness nature of projects an "universal, everywhere applicable and always delivering the correct estimation" technique does not exist, according to Litke [?]. There is also no clear selection progress for the estimation technique to use, beside the time aspect when the use of a technique is possible. It can bee seen in figure 2.2, the function point and the COCOMO technique are both possible after the study stage. As a comparative method, the function point technique has not much in common with COCOMO, which is an algorithmic technique. The project manager has to balance the weigh of each technique and probably choose that one he has most experience with. As an example for estimation techniques these will be described here in more detail with a comparison at the end.

2.2.1 Function Point

The Function Point technique was first mentioned by Allan J. Albrecht in 1979 at the IBM symposium [?]. It was there declared that an useful measurement of productivity is only possible in relation to the functionality that is visible to the user. This measured productivity need to be independent of the used technology and is calculated in with the proportion of project effort and the function points he allocated.
This resulted in to the idea to turn this calculation over for a preliminary estimation of the effort the project would have. According to the clarity and flexibility the technique spread fast. It helps to estimate the scope of a project to an early stage and is suitable for benchmarking in the own company as well as on national or international level [DBP12]. It contains algorithmic and also comparative methods. Basically, this technique uses five steps for estimation [?]:

1. Determining the components
2. Evaluation of the components
3. Calculating the function points
4. Categorization of the influence factors
5. Calculating the development effort with the function points and influence factors

The most important part of this technique is that all measurements only include the user view. This means that the user view is focused on that functions that are important for the specific business process. Implemented business processes are the components that have to be determined.

Determining the Components

To divide the project in components that are useful for the function point estimation, all inherited business processes are divided into elementary process. These

are the smallest and from business perspective view useful, closed activities, that can be performed by the system [DBP12].
A distinction is made into five categories:

1. Input Data
2. Output Data
3. Request
4. Dataset
5. Reference Data

It is useful to categorize the components, because a change in the datasets is followed by more effort than changing a request [?].
The Input data, in literature sometimes called External Inputs (EI), is an elementary process in which data crosses the boundary from outside to inside. This data comes from a data input screen or another application. To maintain one or more logical files, this data can be used for or to control business informations.
External Outputs (EO), or Output Data, are an elementary process in which derived data passes across the boundary from inside to outside. Created output files or reports are sent to other applications. These are created from one or more internal logical files and external interface files.
The next category are requests or External Inquiry (EQ). An elementary process with both input and output components that result in data retrieval from one or more internal logical files and external interface files. This process does not update any Internal Logical files, and the output side does not contain derived data.
Internal Logical Files (ILF's) or Dataset are an user identifiable group of logically related data that resides entirely within the applications boundary and is maintained through external inputs.
The last category are the Reference Data or External Interface Files (EIF), which are a user identifiable group of logically related data that is used for reference purposes only. The data resides entirely outside the application and is maintained by another application [?].
After the categorization of the components there are for each category an amount of components, which then have to weighted for further evaluation.

Evaluation of the Components

The evaluation of the components simply is the classification of each category in their level of difficulty simple, medium or complex. After this, each component is multiplied with the point value according to their difficulty.
The following table is described by Wieczorrek and are standard values for the function point technique [?]:

Category	simple	medium	complex
Input Data	3	4	6
Output Data	4	5	7
Request	3	4	6
Dataset	7	10	15
Reference Data	5	7	10

Table 2.2.1. Point Value of each Category

Calculating the amount of Function-Points

After the evaluation there is an amount of components for each of the Categories from above. To get the number of function points each category has to be multiplied according to the selected weigh and all categories are summed. This results in the following equation:

$$\boxed{\mathit{E1} = \sum_{1}^{n} \mathit{Function} * \mathit{Difficulty}}$$

Function is the respective category and difficulty is the weigh from the table 2.2.1. The resulted value E1 is necessary for evaluating the estimated points with the influence factor.

Classification of Influence Factors

Do get a more realistic estimation, all influences that can affect the project surroundings are measured. For the Function Point estimation there are seven defined influence factors [HB08], which are described below.
Each of the influence factor has value between zero and five which describes how much the factor influences the project.

Integration into other applications
The system will work with different applications and will send and receive data from other applications. This states if there is a cooperation with other applications and if the communication exists online or offline.

Local Data Processing
This factor describes if the system will work with distributed data. Zero means that the system does not work with other applications and five means that there is an integration into other applications in both ways.

Transaction Rate
A high transaction rate affects planing, development, installation and maintenance of the system. It describes how much transactions are to expected with the system.

Processing Logic

The processing logic can be divided in 4 subcategories: Arithmetic Operation, Control Procedure, Exception Regulation and Logic. Arithmetic Operation describes the intensity of the operations in the Project. The controlling of the results is stated within the Control Procedure influence multiplier. With the Exception Regulation is described how eventual exceptions are treated and the Logic multiplier describes how much effort is to be expected for planning the logical component of the project.

Reusability
How much of the produced software has to be reusable in other projects. A high value in re usability means extra effort in the planning stage for module-based development.

Stock Conversion
This describes how much of the used data need to be transformed for use within the project. A high value means much input from other applications that has to be transformed into data the application can process.

Facilitate Change
The application was especially planned and developed that changes could be made easily. A high value means that the user can make changes on the system on its own and that the changes are available immediately.

When all influence factors are set, all values for each factor will be summed up to the value E2.

$$\boxed{E2 = \sum_{1}^{n} \mathit{Influence\ Factor}}$$

This influence factor indicator hast then to be transformed to a multiplier that calculable with function points [HB08][?].

$$\boxed{E3 = \frac{E2}{100} + 0{,}7}$$

Entwicklungsaufwand berechnen

From the calculated Function Points (E1) and the Influence Factor multiplier (E3) are now the Total-Function-Points (TFP) calculated. These TFP are calculated with the following formula:

$$\boxed{TFP = E1 * E3}$$

From the regression analysis of previous project a standard calculation of Function Points per day can be made with the following table:
The project has to be classified with their Total-Function-Points and divided through the appropriate points per day. The result of this calculation are the expected Man-Days for this project, evaluated with the function point technique.

Estimated Size	Function Points	Points per Day
Small Project	till 350	18
Mid Small Project	till 650	16
Medium Project	till 1100	14
Mid Large Project	till 2000	12
Large Project	as of 2000	10

Table 2.2.2. Function Points to Days

2.2.2 COCOMO

Die Cocomoberechnung ist ein Verfahren, dass auf algorithmischen und parametrischen Methoden basiert[70]. Es handelt sich um ein empirisches Modell, bei dem mehrere firmenspezifische Daten (Parameter) von Softwareprojekten zusammenfließen. Durch die Analyse können Formeln abgeleitet werden, welche die Beobachtungen bestätigen. Hierdurch findet eine Verknüpfung der Systemgröße, Produkt-, Projekt- und Teamfaktoren mit dem Aufwand für die Entwicklung des Systems statt[71]. Für den Einsatz des Cocomo-Verfahren sprechen folgende Aspekte: es handelt sich um ein gut dokumentiertes, frei verfügbares Verfahren, das von Public-Domain-Software und kommerziellen Werkzeugen unterstützt wird, weit verbreitet in Organisationen, das Verfahren hat eine lange Historie und wurde über die Jahre an die IT-Entwicklung angepaßt (Das Verfahren wurde bereits 1981 durch Barry W. Boehm, Softwareingenieur bei Boeing, entwickelt) Die Genauigkeit einer mit dem Cocomo-Verfahren angewendeten Schätzung für ein Softwareprojekt wird mit fortschreitender Projektentwicklung immer höher. Beim Start des Projektes kann die Schwankungsbreite noch (x Monaten) zwischen 0,25x bis 4x liegen. Während des Fortschreitens wird die Schwankungsbreite immer kleiner bis sie kurz vor Beendigung des Projektes auf Null sinkt[71]. Das Cocomo-Verfahren umfasst eine große Anzahl von komplexen Parametern, die nicht einfach zu beschreiben sind. Es findet eine Unterteilung in Projektklassen, Modellvarianten sowie Entwicklungszeit- und –aufwände sowie Projektprofilen statt.

Projektklassen

Die Projekte werden in drei Projektkomplexitäten mit verschiedenen Berechnungsfaktoren eingestuft: Projektkomplexität Berechnungs- faktor KDSI Beschreibung Einfach (Organic Mode) 1.05 ¡ 50 ein kleines, gut harmonierendes Team arbeitet zusammen, Umfeld ist bekannt, keine große Innovation notwendig, stabile Schnittstelle, kein Druck durch Endtermin Mittel (Semidetached Mode) 1.12 50 - 300 Mitarbeiter mit durchschnittlicher Erfahrung, erfahrene und weniger erfahrene Projektteam-Mitglieder, Projektteam-Mitglieder mit einigen Erfahrungen in Teilgebieten arbeiten zusammen Komplex (Embedded Mode) 1.20 ¿ 300 starker Kosten- und Termindruck, große Innovation, umfangreiches Produkt mit integralen Elementen, hohe Anforderungen ans PT, neue Komponenten

Modellvarianten

Man unterscheidet zwischen dem Basis-, dem Zwischen- und dem Erweiterten-Cocomo-Verfahren. Bei dem ersten Cocomo-Modell bezieht man sich auf ein Drei-Stufen-Modell. In diesem Modell entsprechen die Stufen der Ausführlichkeit der Analyse einer Schätzung. Bei der ersten Stufe erfolgt eine Basis-Schätzung, eine sog. grobe Schätzung. Mit diesem Basismodell werden überwiegend die Projektkosten in einem frühen Stadium des Projektes nach der Studienphase geschätzt, da der Detaillierungsgrad der Ablaufstruktur noch nicht zu hoch ist. Mittels einer Grundgleichung werden Kosten- und Zeitaufwand errechnet. Das Projekt wird als ein Ganzes betrachtet, ohne eine Unterteilung ins zeitliche und strukturelle vorzunehmen. Der Schwierigkeitsgrad ist konstant gleich hoch. Dieses Basismodell gilt als nützlicher Ausgangspunkt für die nachfolgenden Projektaufwandschätzungen. Mit Hilfe der zweiten Stufe wird die Basis-Schätzung unter Anwendung von Projekt- und Prozessmultiplikatoren verfeinert. Hier wird noch nicht die Differenzierung der Entwicklungsphasen berücksichtigt. Von einem parametrisierten Typ kann noch nicht gesprochen werden, da eventuell nicht belegbare Daten berücksichtigt werden können. Zu guter letzt werden bei dem erweiterten Verfahren die Schätzungen mit den 15 Einflussfaktoren den verschiedenen Phasen zugeordnet[75].

Entwicklungszeit- und aufwand

Die Formel zur Schätzung des Entwicklungsaufwandes bzw. der benötigten Zeit für die Softwareentwicklung des Basismodells entnehmen Sie der unteren Auflistung: Einfache SW-Projekte PM = 2.4 * (KDSI)1,05 Mittelschwere SW-Projekte PM = 3.0 * (KDSI)1.12 Komplexe SW-Projekte PM = 3.6 * (KDSI)1.20

Die Formel zur Schätzung des Entwicklungsaufwandes bzw. der benötigten Zeit für die Softwareentwicklung des Zwischenmodells lautet: Einfache SW-Projekte PM = 3.2 * (KDSI)1,05 Mittelschwere SW-Projekte PM = 3.0 * (KDSI)1.12 Komplexe SW-Projekte PM = 2.8 * (KDSI)1.20

Bei den Berechnungen legt Barry Boehm folgende Parameter fest: ein Personen-Monat besteht aus 152 Arbeitsstunden, bei 19 Arbeitstagen mit 8 Arbeitsstunden pro Tag. Abwesenheiten wie Krankheit, Urlaub und Ferien sind bereits berücksichtigt. Die nachfolgende Abbildung zeigt, dass je größer ein Produkt ist, desto kleiner die Leistung von einem Full-time-equivalent wird. Konkretisiert heißt das, dass bei kleinen Produktgrößen qualifizierte Informatiker mehr Codes schreiben können als bei im Vergleich zu großen Produkten[77]. Die für die Berechnung der Entwicklungszeit benötigte Formel lautet: Einfache SW-Projekte TDEV = 2.5 * (PM)0.38 Mittelschwere SW-Projekte TDEV = 2.5 * (PM)0.35 Komplexe SW-Projekte TDEV = 2.5 * (PM)0.32

Entwicklungszeit- und aufwand

Jenny unterteilt die Projetkgrößen nach der Anzahl der Codezeilen (loc) in folgende Klassen[79]: Small Kleines Projektprofil 2000 loc Intermediate Mittleres

Projektprofil 8000 loc Medium Mittelgroßes Projektprofil 32000 loc Large Großes Projektprofil 128000 loc Very large sehr großes Projektprofil 512000 loc

2.2.3 Comparison

adsf a
a
a
a
a
a
a
a
a

2.3 State of the art

Vergleich welche Software es hierzu auf dem Markt gibt. Kurzbeschreibung der Software, Lizenzmodell, Kosten, Marktrelevanz a
a
a
a
a
a
a
a
a

2.3.1 The market

Wieviele Softwareanbieter gibt es. Ist am Markt eine Nachfrage dafür. Gibt es mobile Anwendungen? Wird nach mobileren Anwendungen verlangt a
a
a
a
a
a
a
a
a

2.3.2 SEER - Cost Estimation Software

The “Seer - Cost Estimation Software” is developed by Galorath Inc. in Los Angeles, USA. Galorath Inc. offers Consulting and Software for Cost Estimation,

Decision Support and Project Management. The company was founded in 1979 with the goal to improve the software and hardware development process in the industry. The next step was to improve their consulting quality by developing their own tools for this process. Seer is the name of their toolset whit a large variety of how the different tools support the development process of new products.
Seer for Software is an estimation application for ”estimating, planning, analyzing and managing complex software projects” (galorath.com/products/software/SEER-Software-Cost-Estimation). Based on the SEER design principles the software contains an annotated and guided interface for defining projects, a parametric simulation engine and numerous standard and custom reporting options. Due to an open architecture API the SEER application can be integrated with enterprise applications and departmental productivity solutions. Galorath specifies that all estimations within this software are repeatable and consistent.
According to the software description, ”a high-level software estimate can be developed in a matter of minutes using SEER´s intuitive, window-based interface” (http://galorath.com/w content/uploads/2014/08/SEERforSoftware2.pdf). To start a new estimation a dialogue guides the user through the process. In the first screen the user has to set project name and decides whether he creates an empty project or starts with a scenario. The scenarios are example projects with some data for starting the estimation. In the next step the user choose the estimation method he wants to use for this project. The estimation methods are subdivided in functional, lines and sizing scale. Another feature of the software is the documentation and export function. All estimations can be exported to Microsoft Project, Microsoft Office, IBM Rational or other 3rd-party software. SEER delivers a huge variety of estimation methods, guided processes and many possibilities for documentation and reporting of all estimations.
The software can be bought in an estimator, project manager and studio version. The estimator version inherits only standard estimation whereas the project manager and studio version allow estimation checking and access to the projects database. The studio version allows also independent crosscheck and verification. The price of each software version is nowhere specified.

2.3.3 PRICE - Cost Estimation Software

PRICE Systems L.L.C. provides agile and accurate estimating solutions. With their head office in Maunt Laurel, USA, and 12 locations worldwide they offer since 1969 estimating acquisitions. Their Software Development Cost Model is one of the oldest and most widely used software parametric models for software development projects. In 2003 PRICE released their Software TruePlanning, which contains methods that estimate the scope, cost, effort and schedule for software projects.
For cost estimation, purchasing efficiency and budget planning PRICE Systems develops ”multi-faceted cost estimating solutions” (http://www.pricesystems.com/en-us/leadership/priceoverview.aspx). Their biggest project is the PRICE Estimating Systems (ESI) Framework that delivers solutions for estimating projects and

cost management. This framework is not specialized for estimating only software projects but also other projects.

The integrated approach to Lifecycle Cost Management contains the PRICE Cost Estimation Framework. It is possible to estimate with all current state of the art cost estimations. The collaboration with other users is also possible as well as sharing the results through the program. This allows faster teamwork and a better estimation output. A data driven method for all estimations allows to compare the estimation with already completed projects. True Mapper allows also estimation output to a specific work breakdown structure or cost element structure for accurate top-down and bottom-up estimate comparisons. Mappings can be stored, retrieved, and modified to keep pace even as program activities change. Beyond specific cost estimating products and services, PRICE Systems offers strategic cost management services. They collaborate with estimators, engineers, project managers, and financial and executive management to design and implement integrated cost management systems that meet the unique challenges in the environment.

There are not given any details about the cost of the software or licensing. This information must specifically enquire by Price Systems.

2.3.4 SLIM Estimate

SLIM Estimate is a cost estimating software developed by Quantity Software Management (QSM), an estimation company with their head office in McLean, USA. (http://qsma.com/slim-estimate/) QSM was founded in 1978 as a software management consultant company for measure, estimate and control projects. Beside their consulting business segment they develop the SLIM Software which contains software for controlling, metrics and estimation.

The SLIM Estimate software provides a flexible cost estimation which align the estimation to the project size and new technologies. The estimation will hereby in the curse of the project enhanced. Integrated schedules offer simple exports of an existing calculation. The software can suggest alternate solutions calculated on past projects.

The homepage got not into detail how this suggestion works. For a new estimating the user can use the SLIM Database and the QSMA knowledge. QSMA owns a huge database with past projects and industry standard estimation. This knowledge can be used for new estimations.

For better use in the company the SLIM Estimate inherits interfaces to MS Office and the possibility export estimations as a web representation. SLIM Estimate delivers hereby a great software package with many functionalities. Especially the access to a large database and the knowledge base of QSM is a real added value to the application. As a result, it is possible in new projects to access the experience of past projects and get the benefit from this experience.

2.3.5 Kinvey

Kinvey is a software developer from Boston whose main business is the development of apps for business. In terms of cost estimates, the company is still very

interesting. Since this company's mobile applications are developed individually for companies they need getting a cost estimate.
Kinvey offers an online estimate for mobile applications with the potential customers can estimate the costs for their application. Here, the estimates are on the side of the cost and the length of time indicated by the customer would have to spend at a natural development, as well as the costs incurred for the customer if Kinvey developed the app for the company. The cost of developing through Kinvey here are always cheaper than a proprietary development.
The cost estimation process is simple. The estimation schedule is a one-page where the user can select the components of his application. The user is guided through some question about features the application should contain. To answer these questions the user can either select one answer or more, which depends on the actual question. After each step the user can see the actual cost estimation of the project. The costs are calculated from the selected components in man days. It is possible to adapt in a further overview these cost estimates by hand again. It is not possible to define and add your own components for the estimation. The individual items that the user can choose are sorted by groups and displayed graphically. Estimating the cost of app development can also be used for your own estimates, but there is no way to export this estimation.

2.3.6 Commonalities

Was hat die Software gemeinsam, welche besonderen Features bei allen aufgefallen, wo ist ein feature das besonders heraussticht.
a
a
a
a
a
a
a
a
aa
a
a
a
a
a
a
a
a

3

Application concept

In diesem Kapitel sollen die Konzepte die in der Anwendung umgesetzt wurden besprochen werden

3.1 Project planning

Herangehensweise an die Planung, Regelmäßige Meetings mit Betreuer, Allgemeine Features von vorhandener Software Analysiert, neue Features Entwickelt, Im Gespräch mit Rock Featurewünsche besprochen, als nächstes Ziele und Anforderungen bestimmt

3.1.1 Objectives

Auflistung der wichtigsten Ziele, Besprechen der Ziele und wofür diese wichtig sind

3.1.2 Requirements

Herangehensweise. Die Schwierigsten Anforderungen heraussuchen und besprechen

3.1.3 Cost Estimation

Schätzung der eigenen Anforderungen mit Function Point

3.2 Architecture

Schaubild zur Architektur der Anwendung, einzelne Punkte besprechen

3.3 Components

Beschreibung in Welche Komponenten die Anwendung aufgeteilt ist.

3.3.1 Database Helper

Alle SQL Befehle werden über diese Klasse gemacht. Muss in allen Klassen die Zugriff auf die Datenbank haben wollen eingebunden werden. Befehle zu großen Teilen sehr Abstrakt gehalten

3.3.2 Influence Factors

Kompontene mit den Einflussfaktoren entsprechend der gewählten Schätzmethode. Summe Aller Faktoren zur Berechnung mit den Punkten der Schätzmethode

3.3.3 Projekt Daten

Speicherung aller benötigten Daten in einer Klasse. Properties und Einflussfaktor in eugener Klasse und als Objekt in Projekt vorhanden

3.3.4 Related Project

Berechung der Relevanz verschiedener Projekte. Eingehen auf die Grundlage hierfür und Beschreibung der Berechnung

3.3.5 Weitere Features

Export, statistic, Help DB, Feedback, Project Filter, Analysis

3.4 Database design

Vorheriges Design der zentralen Datenbank. Wichtig um die Klassen anzupassen und vorher schon mit wichtigen Daten zu füllen

3.4.1 Project Database

Datenbank für alle Projekte, Eigenschaften, Einflussfaktoren

Project Properties

Welche Tabellen gibt es. Wichtige Tabellen, Aufbau der Tabellen und Grund

Influence Factors

Wie die Einflussfaktoren Aufgebaut, was ist der Gedanke dazu?

Projects

Wie sind Projekte gespeichert, Aufteilung in Projekt, Projektdetails und zugehörige Tabellen, wie die Schätzung Organisiert und wie der Zugriff auf die Elemente

3.4.2 Userinformation Database

Datenbank für Spätere Synchronisation und Userinformationen vom Server, Konzept dazu

3.5 User Interface

3.5.1 Projects Overview

Anordnung der Projekte, Wichtige Informationen zum sehen, Filtern und Suchen nach Projekten

3.5.2 Project Creation

Komponente zur korrekten Erstellung von Projekten, Geführte Eingabe, Korrekte Erstellung von Projekten in der DB, Swipe Funktion

3.5.3 Estimate Function Point Project

Aufbau der Schätzung, Umwandlung von Tabelle in App

3.5.4 Influence Factors

Aufbau der EInflussfaktoren, Neue anlengen

3.5.5 Analysis

Wie die Analyse aufgebau und was soll diese bringen?

3.6 Adjusted Estimation Process

4 Implementation

Was zur Implementierung genutzt + Used Libraries

4.1 Project Structure

Aufbau des Android Projektes und Strukturierung

4.2 Use of an exisiting Database in Android

Wie und warum existierende Datenbank in das Projekt eingebunden

4.3 The Database Helper

Größe der Klasse und Grund dafür, Ein Codefragment besprechen

4.4 Using multilingual Strings with the Databse and Resource Files

Hintergrundgedanke, Wie Funktioniert es, Beispiel Aufruf

4.5 Calculate Person Days

Wie läuft die Kalkulation ab

4.6 Find related Projects

Wie ist der Algorithmus implementiert

4.7 ListView Elements and ViewHolder

Schwierigkeit bei Listviews, Lösung mittels dem ViewHolder besprechen

5

Software Test

Ablauf/Aufbau der Tests

5.1 Test Cases

Beschreibung eines Testfalls, anzahl testfälle, wozu die Testfälle

5.2 User Review

Meinungen von Testern, Verbesserungspotential

6

Conclusion

sdfsdsdff

6.1 Results

Es ist möglich dadurch mobile Schätzungen zu machen, stabile App und verbessert stetig die Schätzungen neuer Projekte, Ohne Vorkenntnisse ist es schwierig die Anwendung zu Benutzen da der Nutzer nicht weiß wie die Schätzungen funktionieren

6.2 Retrospection

Probleme die Schätzung Nutzerfreundlich zu gestalten, Problematik mit dem Projektvergleich war ein schwieriges Problem

6.3 Outlook

Implementierung weiterer Schätzmethoden, weitere Features, Anbindung an einen Server, Anpassung auf Tablets, WebTool

References

ISBSG10. ISBSG : *Practical Software Project Estimation: A Toolkit for Estimating Software Development and Duration.* Mcgraw-Hill Education - Europe, 2010

OH11. OLIVER HUMMEL : *Aufwandsschätzungen in der Software- und Systementwicklung kompakt (IT kompakt).* Spektrum Akademischer Verlag, 2011.

DBP12. DR. BENJAMIN POENSGEN: *Function-Point-Analyse: Ein Praxishandbuch* . dpunkt.verlag, 2012.

MG11. MATTHIAS GEIRHOS: *IT-Projektmanagement: Was wirklich funktioniert - und was nicht (Galileo Computing)* . Galileo Computing, 2011.

SM06. STEVE MCCONNELL : *Software Estimation: Demystifying the Black Art: The Black Art Demystified (Best Practices (Microsoft)).* Microsoft Press, 2006.

HB08. HELMUT BALZERT : *Lehrbuch der Softwaretechnik: Softwaremanagement.* Spektrum Akademischer Verlag, 2008.

RH04. ROBERT HÜRTEN: *Function-Point Analysis - Theorie und Praxis: Die Grundlage für das moderne Softwaremanagement (Edition expertsoft).* expert, 2004.

LMC04. LUIS M. CAMARINHA-MATOS: *Evaluating a Software Costing Method Based on Software Features and Case Based Reasoning.* expert, 2004.

PDM16. PROFESSOR DR.-ING. DAGMAR MEYER : *Projektmanagement - Aufwandsschätzung.* Ostfalia - Hochschule für angewandte Wissenschaften. *(Stand: 04.02.2016)*

FW16. FLORIAN WEIGAND : *Aufwandsschsatzung in IT-Grosprojekten - Function Point Methode.* TU München. *(Stand: 04.02.2016)*

LM16. LEE FISCHMANN: *Evolving Function Points.* Galorath Inc. *(Stand: 04.02.2016)*

LM14. OMG: *Automated Function Points (AFP).* OMG - Object Management Group *(Stand: 04.02.2016)*

A

Erklärung der Kandidatin / des Kandidaten

□ Die Arbeit habe ich selbstständig verfasst und keine anderen als die angegebenen Quellen- und Hilfsmittel verwendet.

□ Die Arbeit wurde als Gruppenarbeit angefertigt.

Datum	Unterschrift der Kandidatin / des Kandidaten